UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/706,076	11/03/2000	Richard A. Willems	PD99-2788	6153

22879 7590 10/20/2004

HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER	
STEELMAN, MARY J	
ART UNIT	**PAPER NUMBER**
2122	

DATE MAILED: 10/20/2004

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	09/706,076	WILLEMS, RICHARD A.	
	Examiner	Art Unit	
	Mary J. Steelman	2122	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *16 July 2004*.

2a)☒ This action is **FINAL**. 2b)☐ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-8 and 10-17* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-8 and 10-17* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *03 November 2000* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application (PTO-152)
6) ☐ Other: _____.

DETAILED ACTION

1. This office action is in response to amendment filed 16 July 2004. Per Applicant's request, claims 1, 6, and 13 have been amended. Claim 9 has been canceled. Claims 1-8 and 10-17 are pending.

Claim Rejections - 35 USC § 103

2. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

3. Claims 1-8 and 10- 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over U.S. Patent 6,158,045 to You, in view of "Compilers Principles, Techniques, and Tools" by Alfred V. Aho, Ravi Sethi and Jeffrey D. Ullman, pages 432-433, 439, and 703-711, and further in view of U.S. Patent Application 2003/0200397 to McAllister et al.

You disclosed symbolic debugging services utilizing a client debugger object, a connection object and a server debugger object. (Abstract, lines 9-11) "An addressing abstraction is utilized to facilitate the use of target memory addresses..." The client debugger object transmits debug requests to a target server debugger object. "Figure 4, the collection represents a possible layout for the memory locations in which pointers to the objects of type Aobject are stored...(col. 11, lines 42)."

Per claims 1 and 11:

-suspending execution of all threads executing on the computer system except for a thread performing the method;

(You: "Breakpoints are specified by process, thread, ad address. They can have several conditions which describe what causes a program to stop, and they can describe what kind of behavior should occur to individual threads and threads in a process when the breakpoint is reached", col. 41, lines 28-30, "kConditionalBreak indicates a breakpoint which has a condition associated with it and should only stop when the condition has been met (for example, a memory access)", col. 41, lines 48-50, "...kStoppinBreak which indicates that when one thread stops, all threads stop in the target process", col. 64, lines 30-32, "The TdebuggerPrimitiveClient object provides several functions which are directed at accessing or controlling information about individual processes", col. 65, lines 1-5, "To kill a thread that is currently executing, the client debugger calls KillThread. Restarting a stopped thread is achieved by calling RunThread...", col. 86, lines 37-43, "The software exception handling policy can be changed to allow the program to stop once a software exception has occurred. The software exception turns into a hardware exception (a breakpoint), and the target program stops, transferring control to the debugger server..." You disclosed suspending execution of all threads except the thread performing the method for gathering data from memory. Suspending processes is well known in the art. It prevents a value from being modified while a debug process is attempting to retrieve the value. Semaphores, monitors, locks, mutex, etc. are used to achieve this suspension. Processes are re-initiated after the lock is no longer needed.)

-gathering data specified by the plurality of memory element descriptors; formatting the data into a buffer. (You: Col. 6, lines 20-22, "A program built on top of a primitive debugger can exploit the capabilities of the debugger to gather dynamic information about the program.")

-restarting execution of the suspended threads after gathering the data;
(You: Col. 65, lines 3-5, "Restarting a stopped thread is achieved by calling RunThread...")

You failed to disclose "after suspending execution of the thread, following a plurality of memory element descriptors of a machine readable record list to locate data in the memory of the computer system, where each memory descriptor is descriptive of data to be retrieved from memory of the computer system"

Although You discloses that memory is accessed, he failed to disclose that memory can be structured as linked lists, and linked lists of linked lists (binary tree). However, Aho disclosed the use of linked lists in memory (page 432-433) and information retrieval from the nodes (page 439). Official Notice is given that binary trees are special forms of linked lists whereby the first linked list can contain the head of a second linked list. Traversal to access data (following a plurality of memory element descriptors) contained in the nodes of data structures and data accessing are well known.

Neither You nor Aho provide extensive details related to "gathering data ...while maintaining data coherency". While You did suggest the stopping and re-starting of processes,

McAllister disclosed additional memory transaction coherency through the use of a memory controller agent (Page 3, [0022]). McAllister disclosed, at page 3, [0023], "The agent is responsible for ensuring coherency and fulfilling memory transactions for a single memory line, thereby simplifying the design of the agent."

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to include information regarding binary trees as useful data structures in memory for storage and retrieval of information because linked lists are useful for creating arrays of unknown size, using non contiguous regions of memory, and retrieving data does not remove or destroy the item and furthermore to include details on data coherency when retrieving data from memory because fresh data, not stale, "dirty" data is necessary when attempting to debug memory. And furthermore, to modify You and Aho to include additional details related to maintaining data coherency because it is important that a data value retrieved truly represents the state at a known point in time. Enforcing data coherency provides this feature. Semaphores, monitors, locks, mutex, etc. are used to achieve this suspension, thereby enforcing data coherency. Processes are re-initiated after the lock is no longer needed.

Per claim 2: (You: Fig. 4 and col. 11, line 32, col. 13, lines 12-13 and 22-23, "...a possible layout for the memory locations...")

Per claim 3: (You: Fig. 5, #511 and col. 55, lines 55-61, "The scalar types long, short, char, signed char, unsigned l9ng...are read and written transparently...Arbitrary blocks of memory and char strings are transported correctly.")

Per claim 4: (You: Col. 14, lines 34-36, "Retrieval from the collection has a uniform, polymorphic interface. This is achieved by calling the Get function which takes an object of the parameterized type.")

Per claims 5 and 12:

You failed to provide details related to "a list memory descriptor" However, Aho disclosed (Page 446-449), "A Fortran compiler can create a number of data areas, i.e., blocks of storage in which the values of objects can be stored..." For each data area the compiler creates a memory map..."

Therefore it would have been obvious, to one of ordinary skill in the art, at the time of the invention to include the teachings of Aho to modify You because Aho is merely reciting well known memory arrangements. Accessing memory locations is well known in the art.

Per claim 6:

You disclosed:

-stopping execution of all threads executing on the computer system except of a thread parsing the list; (You: Col. 65, lines 1-2, "To kill a thread that is currently executing...", col. 86, lines 41-42, "...the target program stops, transferring control to the debugger server...")

-resuming execution of all threads stopped during the step of stopping execution.
(You: Col. 65, lines 3-4, "Restarting a stopped thread is achieved by calling RunThread...")

-extracting data from the node. (You: Col. 6, lines 20-30, "...exploit the capabilities of the debugger to gather dynamic information about the program...")

You disclosed (col. 6, lines 2-4), "A debugger, or interactive program debugger, is a programming tool which is instrumental in allowing a programmer to inspect and control the execution (by inputting a list of data to be retrieved) of a program." You failed to disclose that memory can be structured as linked lists, and linked lists of linked lists (binary tree), all which are commonly traversed to access data. However Aho provided more details:

-constructing a record list, the record list comprising at least a first list element descriptor descriptive of data to be retrieved from a first linked list;
(Aho disclosed the use of linked lists in memory (page 432-433) and information retrieval from the nodes (page 439). Official Notice is given that binary trees are special forms of linked lists whereby the first linked list can contain the head of a second linked list. Traversal to access data

(following a plurality of memory element descriptors) contained in the nodes of data structures and data accessing are well known.)

-following a list head locator of the list element descriptor to a head of the first linked list;
(Official Notice is given that binary trees are special forms of linked lists whereby the first linked list can contain the head of a second linked list. Traversal to access data (following a plurality of memory element descriptors) contained in the nodes of data structures and data accessing are well known.)

-following links of the head of the first linked list to a first node of the linked list;
(Traversal to access data (following a plurality of memory element descriptors) contained in the nodes of data structures and data accessing are well known.)

-interpreting at least one tag of the first list element descriptor to locate data of the node;
(Tag information attached to each node of a linked list is well known in the art.)

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to modify You's disclosure, to include more well known details regarding the arrangement and traversal of memory data structures, as this is well known in the art.

Per claims 7 and 12: (Official Notice that binary tree: node of first linked list contains a head of the second linked list are well known, and thus obvious.)

Per claim 8: (You: Col. 8, lines 45-65, "...Multiple threads can execute concurrently or through simulated concurrency..." "...multiple processes and threads can be executing under the control of a single debugger." "Multithreaded programs can be debugged so that some threads may be stopped and others remain executing while the debugger is also executing."

Per claim 10: (Aho: Page 439, "A pointer front points to the most recently created entry I the list." And "The implementation of lookup is done by scanning the list starting at the entry pointed to by front and following links until the desired name is found...")

Therefore, it would have been obvious, to one of ordinary skill in the art, at the time of the invention, to include details regarding memory access as disclosed by Aho, because this is well known in the art.

Per claim 13:

-a collection driver for execution on the target machine; (You: col. 6, lines 38-43, "Interactive program debuggers...")

-a user interface capable of coupling to the collection driver; (You: Col. 6, line 43, "...graphical user interfaces.")

-a symbol resolution system capable of coupling to the user interface; (You, col. 6, lines 45-52, "...view his program through the representation of program symbols. The program symbols are typically the names of subroutines, classes, types, variables, and other program constructs..."

-wherein the user interface comprises computer readable code for constructing an input record list containing records describing data to be captured, at least some records of the input record list containing information derived from symbols resolved by the symbol resolution system, and transmitting the input record list to the collection driver; (You: Col. 6, lines 20-30.)

-wherein the collection driver further comprises code for stopping execution of all threads executing on the computer system except the collection driver; (You: Col. 65, lines 1-2, "To kill a thread that is currently executing, the client debugger calls KillThread.")

-wherein the collection driver further comprises code for interpreting the input record list and collecting operating system data into a capture buffer specified by the input record list, and transmitting the capture buffer to the user interface. (You: Col. 6, lines 1-52.)

-wherein the collection driver further comprises code for resuming execution of all previously-stopped threads. (You: Col. 65, lines 3-4, "Restarting a stopped thread is achieved by calling RunThread.")

Per claim 14: (You: Col. 6, lines 20-22, "A program built on top of a primitive debugger can exploit the capabilities of the debugger to gather dynamic information about the program...")

Per claim 15: (Official Notice: Retrieval of data from nodes of a tree are well known. Also see You, figure 11, preparing a client request.)

Per claim 16: (You: Col. 55, lines 61-63, "The PDS streams provide a simple abstraction that allows objects to be moved from the writer of the stream to the reader of the stream." And col. 55, lines 55-60, "The scalar types ...are read and written transparently..." And col. 55, lines 62-63, "...a buffered streaming class is provided..." Also see You, figure 5.)

Per claim 17: (You: Abstract, line 9, "Clients can process locally and remotely.")

Response to Arguments

4. **Applicant has argued, in substance, the following:**

(A) As Applicant has pointed out on page 9, 4th paragraph, of Amendment dated 16 July 2004, the cited references fail to teach "suspending the execution of threads executing on a computer system to maintain data coherency."

Examiner's Response:

The claim limitation "suspending execution of all threads executing on the computer system except for a thread performing the method...after suspending execution of the thread...restarting execution of the suspended thread after gathering the data..." is inherent in data collecting. See rejection of claim 1 above: "Breakpoints are specified by process, thread, ad address. They can have several conditions which describe what causes a program to stop, and they can describe what kind of behavior should occur to individual threads and threads in a process when the breakpoint is reached", col. 41, lines 28-30, "kConditionalBreak indicates a breakpoint which has a condition associated with it and should only stop when the condition has been met (for example, a memory access)", col. 41, lines 48-50, "...kStoppinBreak which indicates that when one thread stops, all threads stop in the target process", col. 64, lines 30-32, "The TdebuggerPrimitiveClient object provides several functions which are directed at accessing or controlling information about individual processes", col. 65, lines 1-5, "To kill a thread that is currently executing, the client debugger calls KillThread. Restarting a stopped thread is achieved by calling RunThread...", col. 86, lines 37-43, "The software exception handling policy can be changed to allow the program to stop once a software exception has occurred. The software exception turns into a hardware exception (a breakpoint), and the target program stops, transferring control to the debugger server..." You disclosed suspending execution of all threads except the thread performing the method for gathering data from memory. Suspending processes is well known in the art. It prevents a value from being modified while a debug process is attempting to retrieve the value. . Suspension of threads is well known through the use of semaphores, monitors, lock, sleep, etc. Processes are re-initiated after the lock is no longer needed. You: Col. 6, lines 20-22, "A program built on top of a primitive debugger can exploit

the capabilities of the debugger to gather dynamic information about the program.", (col. 65, lines 3-5), "Restarting a stopped thread is achieved by calling RunThread...")
Additionally, McAllister disclosed, at page 3, [0023], "The agent is responsible for ensuring coherency and fulfilling memory transactions for a single memory line, thereby simplifying the design of the agent." McAllister reinforces the well known fact that accurate collection of a data value from memory requires data coherency.

Examiner maintains the rejections of claims 1-8 and 10-17.

Conclusion

5. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Mary Steelman, whose telephone number is (703) 305-4564. The examiner can normally be reached Monday through Thursday, from 7:00 AM to 5:30 PM If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam can be reached on (703) 305-4552. The fax phone number for the organization where this application or proceeding is assigned is 703-872-9306.

After October 25, 2004, examiner can be reached at new telephone number (571) 272-3704. Supervisor, Tuan Q. Dam can be reached at (571) 272-3694.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mary Steelman

10/05/2004

ANTONY NGUYEN-BA
PRIMARY EXAMINER